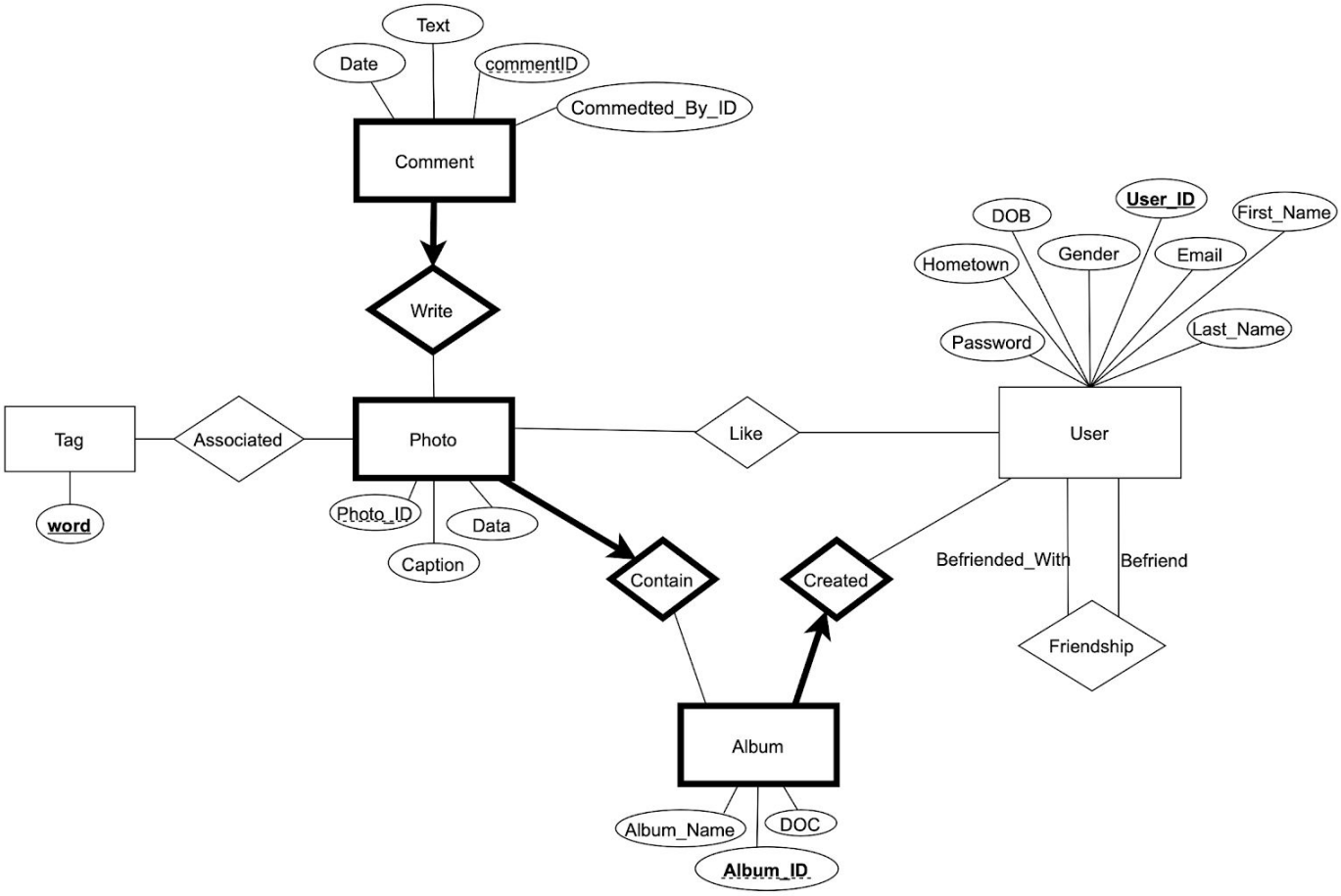
CS460 #PA1
Group Member: Tom Yu & Mingjun Wen



Assumptions & constraints in our design:

- Each user has an unique User ID. The friend of an user is also an instance of the User entity. Therefore each friend of the user has an unique User ID.
- Although email is not the primary key of the User entity, it is a required attribute since each user must have an unique email address upon registration.
- Users with the same User_ID can be friends since a user cannot be friends with him/herself, each user must have an unique Email.
- For Photo entity and Tag entity, many photos can have the same tag and a tag can have many photos associated with it.
- If the user deletes the album, all photos inside the album will also be deleted. Therefore Photo is a weak entity of Album.
- If a Photo is deleted, all comments under the photo will be deleted. For this reason we will implement Comment as a weak entity of Photo.
- In the Comment entity, there is an attribute called Commedted_By_ID. It is referencing the User_ID of the user who created the comment. However, Commedted_By_ID can be NULL, since people can comment anonymously.
- In Photo entity, there is an attribute called Data, which is used to store the file location (path) of the image. This location is assumed to be unique since it's impossible for two files to have the same location.
- Like is a many to many relation between User and Photo. We will use basic relational algebra to count the number of likes of each post and to show which user liked which photo.
- Other functionalities such as Contributions, Friend Recommendations, and Tags ranking will be implemented using relational algebra when we create the software. We had made sure that the diagram included all the necessary information to implement those functions. Therefore, these functionalities might not directly be reflected in our ER diagram.

Integrity Constraint:

- In a Friendship relation, user1_ID and user2_ID cannot be the same due to the fact that a person cannot be friend to him/herself.

```
CREATE TABLE User(
        User_ID CHAR(10) NOT NULL,
        First_Name CHAR(50),
        Last_Name CHAR(50),
        DOB INTEGER,
        Email VARCHAR(30) NOT NULL,
        Password VARCHAR(15),
        Gender CHAR(10),
        Hometown CHAR(50)
        PRIMARY KEY (User_ID)
);


CREATE TABLE Album(
        User_ID CHAR(10) NOT NULL,
        Album_ID CHAR(15),
        Album_Name CHAR(50),
        DOC INTEGER,
        PRIMARY KEY (User_ID, Album_ID),
        FOREIGN KEY (User_ID) REFERENCES (User)
                ON DELETE CASCADE
);


CREATE TABLE Photo(
        User_ID CHAR(10) NOT NULL,
        Album_ID CHAR(15) NOT NULL,
        Photo_ID CHAR(20),
        Caption CHAR(150),
        Data VARCHAR(128),
        PRIMARY KEY (User_ID, Album_ID, Photo_ID),
        FOREIGN KEY(User_ID) REFERENCES User(User_ID)
                ON DELETE CASCADE,
        FOREIGN KEY(Album_ID) REFERENCES Album(Album_ID)
                ON DELETE CASCADE
);


CREATE TABLE Tag (
        word CHAR(20) PRIMARY KEY
);
```

```
CREATE TABLE Comment(
        Text CHAR(50),
        Date INTEGER,
        Commedted_By_ID CHAR(10),
        User_ID CHAR(10) NOT NULL,
        Album_ID CHAR(15) NOT NULL,
        Photo_ID CHAR(20) NOT NULL,
        Comment_ID CHAR(25),
        PRIMARY KEY(User_ID, Album_ID, Photo_ID, Comment_ID),
        FOREIGN KEY (User_ID) REFERENCES User(User_ID)
                ON DELETE CASCADE,
        FOREIGN KEY (Album_ID) REFERENCES Album(Album_ID)
                ON DELETE CASCADE,
        FOREIGN KEY (Photo_ID) REFERENCES Photo(Photo_ID)
                ON DELETE CASCADE,
        FOREIGN KEY(Commedted_By_ID) REFERENCES User(User_ID)
);


CREATE TABLE Friendship(
        userID1 CHAR(10) NOT NULL,
        userID2 CHAR(10) NOT NULL,
        PRIMARY KEY(userID1, userID2),
        FOREIGN KEY (userID1) REFERENCES User(User_ID),
        FOREIGN KEY (userID2) REFERENCES User(User_ID),
        CHECK(userID1 <> userID2)
);

CREATE TABLE Like(
        User_ID CHAR(10),
        Photo_ID CHAR(20),
        PRIMARY KEY (User_ID, Photo_ID),
        FOREIGN KEY (User_ID) REFERENCES User(User_ID),
        FOREIGN KEY (Photo_ID) REFERENCES Photo(Photo_ID)
);
```

```
CREATE TABLE Associated(
        User_ID CHAR(10) NOT NULL,
        Album_ID CHAR(15) NOT NULL,
        Photo_ID CHAR(20),
        word CHAR(20),
        PRIMARY KEY (word, User_ID, Album_ID, Photo_ID),
        FOREIGN KEY (User_ID) REFERENCES User(User_ID),
                ON DELETE CASCADE,
        FOREIGN KEY (Album_ID) REFERENCES Album(Album_ID),
                ON DELETE CASCADE,
        FOREIGN KEY(Photo_ID) REFERENCES Photo(Photo_ID),
        FOREIGN KEY (word) REFERENCES (Tag)
);
```